COLLEGE OF SOCIAL & BEHAVIORAL SCIENCES
School of Information

Layer 8+ Privacy: Biometrics I

Information Privacy with Applications

David Sidi (dsidi@email.arizona.edu)

Warm-up

- Summarize some part of what Dan Geer said in Identity as Privacy

Small mention of interesting things

- In the VM, the command line uses ksh with vim bindings
  - demonstration
  - if you don’t like this, you can use bash
- Arbitrary code execution on the Intel ME (link)

(finishing last time)
Trust: Counsel of Despair?

Reflections on Trusting Trust

the simplest quine

the simplest quine

careful! The empty program is copyrighted

/bin/true

```
#
#     Copyright (c) 1984 AT&T
#       All Rights Reserved
#
#     THIS IS UNPUBLISHED PROPRIETARY SOURCE CODE OF AT&T
#     The copyright notice above does not evidence any
#     actual or intended publication of such source code.
#
#ident        "@(#)cmd/true.sh        50.1"
```

A simple Quine

```
char s[ ] = {
        '\t',
        '0',
        '\n',
        '}',
        ';',
        '\n',
        '\n',
        '/',
        '*',
        '\n',
        (213 lines deleted)
        0
};

/*
 * The string s is a
 * representation of the body
 * of this program from '0'
 * to the end.
 */

main( )
{
        int i;

        printf("char\ts[ ] = {\n");
        for(i=0; s[i]; i++)
                printf("\t%d, \n", s[i]);
        printf("%s", s);
}
```

Here are some simple transliterations to allow a non-C programmer to read this code.

=	assignment
==	equal to .EQ.
!=	not equal to .NE.
++	increment
'x'	single character constant
"xxx"	multiple character string
%d	format to convert to decimal
%s	format to convert to string
\t	tab character
\n	newline character

FIGURE 1.

A simple quine

(not idiomatic)

```
char s[ ] = {
        '\t',
        '0',
        '\n',
        '}',
        ';',
        '\n',
        '\n',
        '/',
        '*',
        '\n',
        (213 lines deleted)
        0
};

/*
 * The string s is a
 * representation of the body
 * of this program from '0'
 * to the end.
 */

main( )
{
        int i;

        printf("char\ts[ ] = {\n");
        for(i=0; s[i]; i++)
                printf("\t%d, \n", s[i]);
        printf("%s", s);
}
```

Here are some simple transliterations to allow a non-C programmer to read this code.

=	assignment
==	equal to .EQ.
!=	not equal to .NE.
++	increment
'x'	single character constant
"xxx"	multiple character string
%d	format to convert to decimal
%s	format to convert to string
\t	tab character
\n	newline character

FIGURE 1.

A quine in Python

```
from string import Template

second = Template('print(${sq}from string import Template${sq})\nprint("second =
Template(${sq}" + second.template.encode(${sq}unicode_escape${sq}).decode() + "$
{sq}")\nprint(second.substitute(sq="${sq}"))')

print('from string import Template')

print("second = Template('" + second.template.encode('unicode_escape').decode()
+ "'")

print(second.substitute(sq="'"))
```

This a quick one I hacked together. There are (much) simpler ones, more boring ones, more interesting ones. Try it!

```
...
c = next( );
if(c != '\\')
        return(c);
c = next( );
if(c == '\\')
        return('\\');
if(c == 'n')
        return('\n');
if(c == 'v')
        return('\v');
...
```

FIGURE 2.1.

```
...
c = next( );
if(c != '\\')
        return(c);
c = next( );
if(c == '\\')
        return('\\');
if(c == 'n')
        return('\ n');
if(c == 'v')
        return(11);
...
```

FIGURE 2.3.

quines in Compilers

targeting login command

```
compile(s)
char *s;
{
        ...
}
```

FIGURE 3.1.

```
compile(s)
char *s;
{
        if(match(s, "pattern")) {
                compile("bug");
                return;
        }
        ...
}
```

FIGURE 3.2.

```
compile(s)
char *s;
{
        if(match(s, "pattern1")) {
                compile ("bug1");
                return;
        }
        if(match(s, "pattern 2")) {
                compile ("bug 2");
                return;
        }
        ...
}
```

FIGURE 3.3.

emit a backdoor!

“that’s worrying... let’s audit the source code for both the login command and for the compiler we use, and recompile everything cleanly”

quines in Compilers

```
compile(s)
char *s;
{
        ...
}
```

FIGURE 3.1.

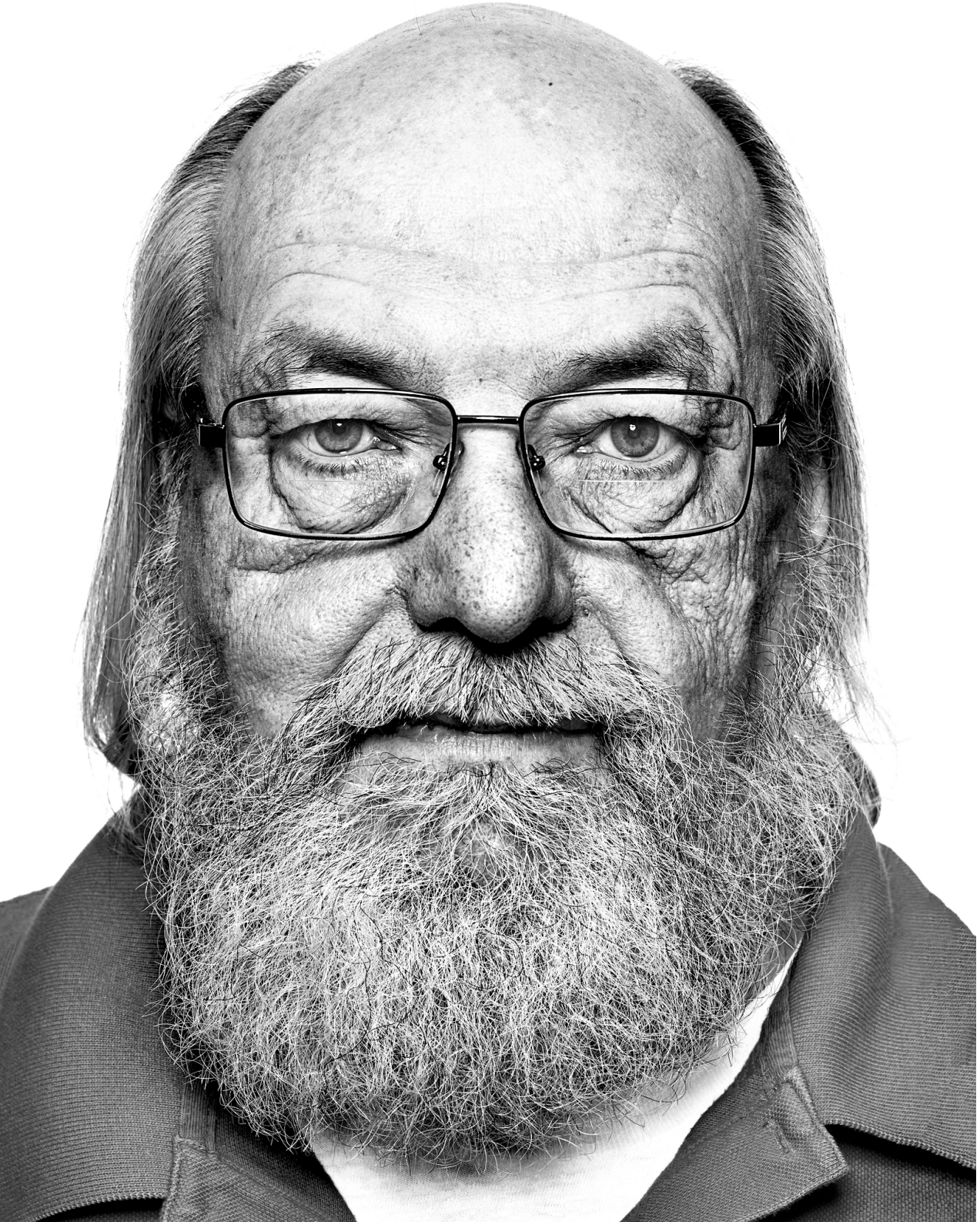
```
compile(s)
char *s;
{
        if(match(s, "pattern")) {
                compile("bug");
                return;
        }
        ...
}
```

FIGURE 3.2.

```
compile(s)
char *s;
{
        if(match(s, "pattern1")) {
                compile ("bug1");
                return;
        }
        if(match(s, "pattern 2")) {
                compile ("bug 2");
                return;
        }
        ...
}
```

FIGURE 3.3.

targeting compiler



The moral is obvious. You can't trust code that you did not totally create yourself. (Especially code from companies that employ people like me). No amount of source-level verification or scrutiny will protect you from using untrusted code.

Ken Thompson, ACM Turing Award Speech, “Reflections on Trusting Trust”

Is Ken Thompson right? (2 min)

Is Ken Thompson right?

- Narrow answers:
  - Rewrite the compiler completely
  - Wheeler’s Diverse Double Compiling (link)

Is Ken Thompson right?

- Narrow answers:
  - Rewrite the compiler completely
  - Diverse Double Compiling
- Broader answers (revealed by narrow ones):
  - Pretty much: yes. You can’t trust **code** that you did not **totally** create yourself, from compiler (or even microcode) on up

Is Ken Thompson right?

- Narrow answers:
  - Rewrite the compiler completely
  - Diverse Double Compiling
- Broader answers (revealed by narrow ones):
  - Pretty much: yes. You can’t trust **code** that you did not **totally** create yourself, from compiler (or even microcode) on up
  - for mortals, that’s all code
  - You need to trust people. Layer 8+ is inescapable.

Example Technologies at layer 8

- PGP web of trust
- ceremonies for verifying keys in instant messengers
- biometrics countermeasures
- shoulder surfing countermeasures
- reputation systems
- CGMS-A

What can happen if you don’t use the PGP web of trust?

- https://stallman.org/gpg.html
- Or even if you do?
  - Ian Goldberg’s shadow attack @ 15:59- 20:06
  - “hilarity ensues”
- What could be done about this attack? (2 min)

Biometrics

Biometrics before computers

- handwritten signatures
- facial features
- fingerprints
- hand geometry
- seals
- pronunciation

- ... mostly not rigorous; imprecise

Biometrics with computers

“Technical progress in image acquisition guarantees observability now; technical progress in standoff biometrics guarantees identifiability real soon now. Venture capitalists regularly hear new ideas in standoff biometry, and Moore’s law is its friend ...

“We will soon live in a society where identity is not an assertion (“Call me Dan”) but rather an observable (“Sensors say that’s Dan”). Your breath, the microwave emissions of your beating heart, the idiosyncratic anomalies of how you type, talk, and walk say who you are.”

Dan Geer, Identity as Privacy

“Moore’s law is its friend … . We will soon live in a society where identity is not an assertion (“Call me Dan”) but rather an observable (“Sensors say that’s Dan”).”

What does this mean, and why might it be important?

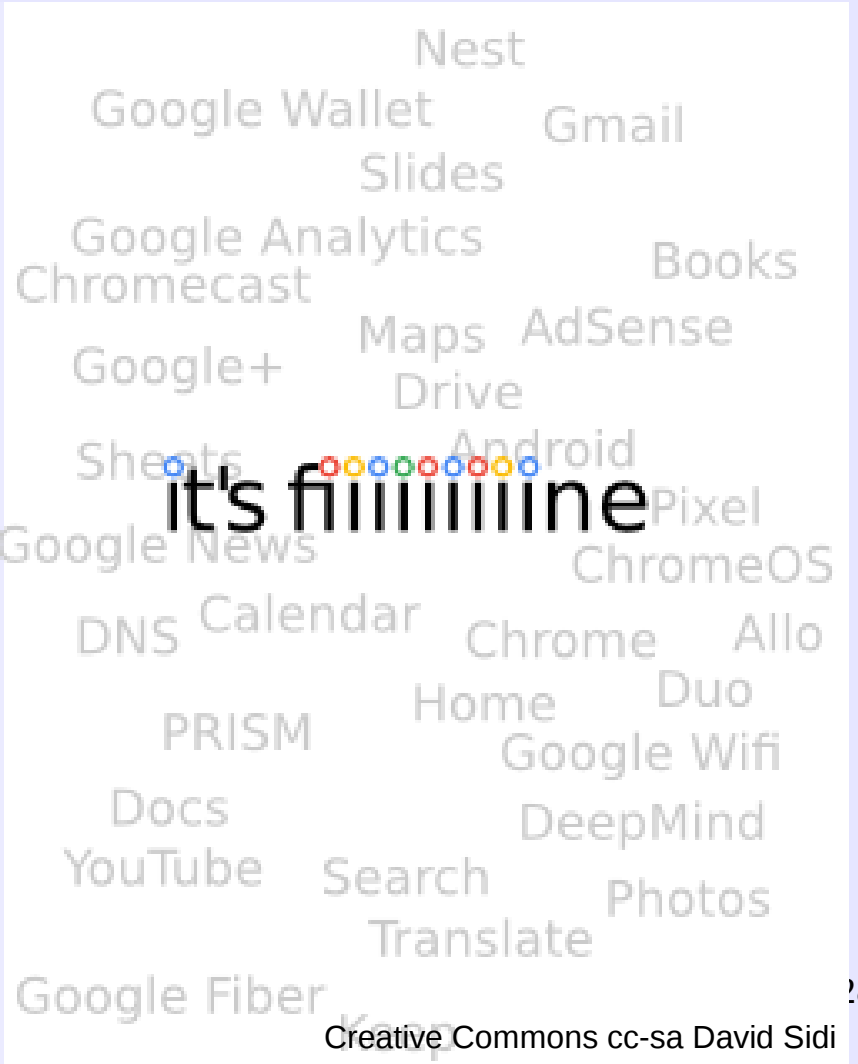
Standoff biometrics are used by law enforcement

- The Police Department for the city of San Diego uses facial-recognition technology to attempt real-time identification of every face that passes in front of a camera
- The FBI's Next-Generation Identification (NGI) searches 16 states' driver's license databases, American passport photos, and photos from visa applications, amounting to nearly 412 million records

  https://www.perpetuallineup.org

Companies are using standoff biometric identification too

- Google's photo service organizes photo albums by recognizing who is photographed
- Google links this biometric data to data from its more than 60 services

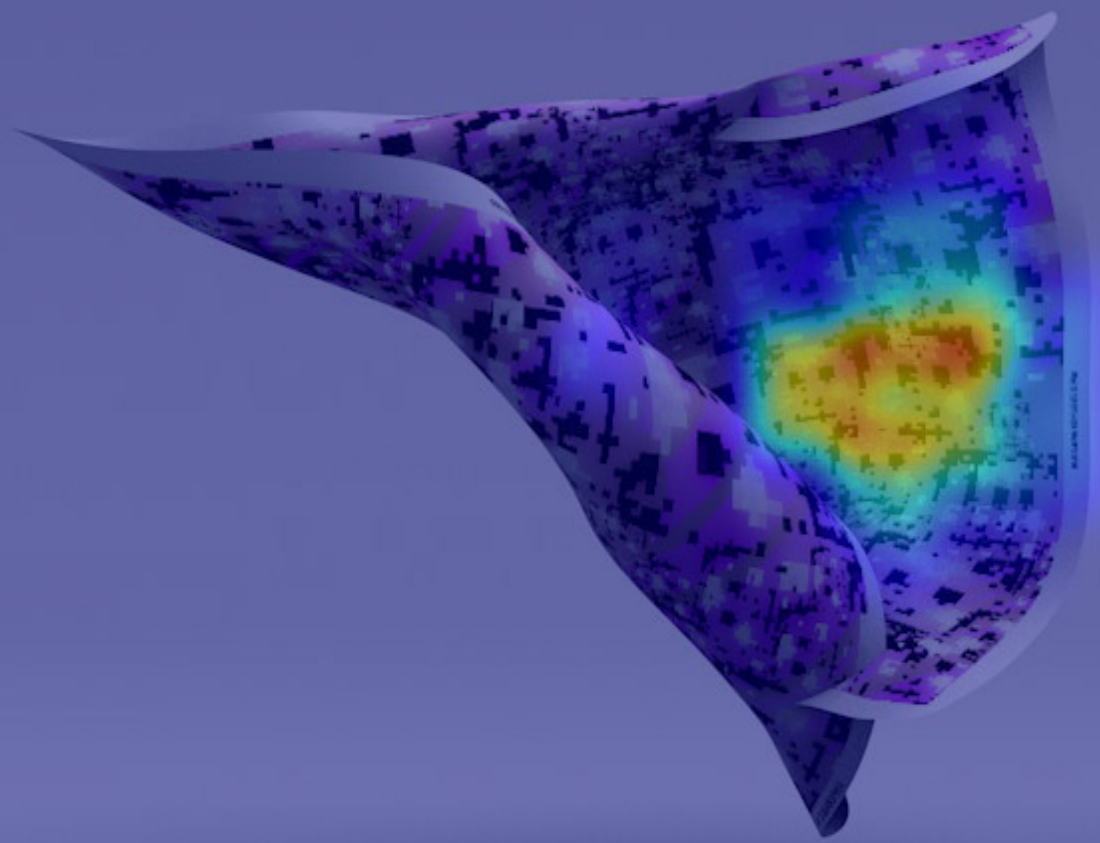


Creative Commons cc-sa David Sidi

“In identity verification, the subject looks straight at the camera under controlled lighting conditions, and their face is compared with the one on file. A related but harder problem is found in forensics, where we may be trying to establish whether a suspect’s face fits a low-quality recording on a security video. The hardest of all is surveillance, where the goal may be to scan a moving crowd of people at an airport and try to pick out anyone who is on a list of perhaps a few hundred known suspects.” (Biometrics, 265)

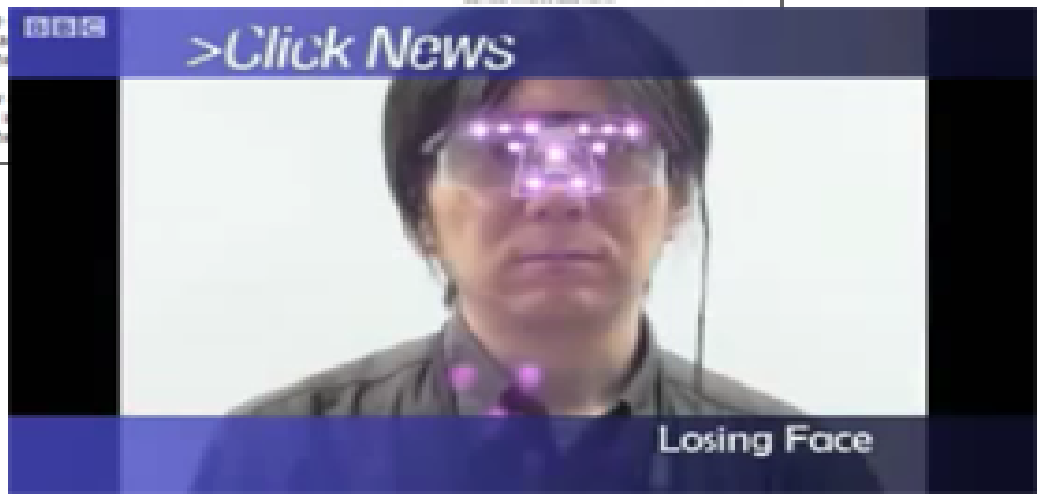
All this for 10% decrement in performance?
https://link.springer.com/chapter/10.1007/978-3-319-41501-7_38

HyperFace™



HyperFace™ Type-01 Prototype / Adam Harvey

Rendering by Ece Tankal / Hyphen-Labs



取り上げられたメディアは
TIME, BBC, NBC, ABC,
NY Times, Spiegel,
ACM Tech Newsなど
海外300以上。
海外でも待ち望まれています！

Figure 1: Can you see the face in each of these images? ***Facebook's automatic face detector can detect and localize all six faces shown above****, even when we try to hide the face by (A) adding occluding noise, (B) adding distortion, (C) blurring the face region, or (D) altering image lighting. Deliberate countermeasures such as (E) wearing a "privacy visor" with infrared LEDs [24] or (F) wearing "Dazzle"-style makeup [9] are not always effective—sometimes Facebook can see through these disguises too. Facebook may not be able to* recognize *these faces, but if Facebook can* detect *them, it may prompt friends to tag these hard faces and reveal their identity.*

Wilbur et al., "Can we still avoid automatic face detection?"

Figure 8: All 30 missed faces in the COFW training set.

Tricking state of the art face detection

Disguised Face Identification

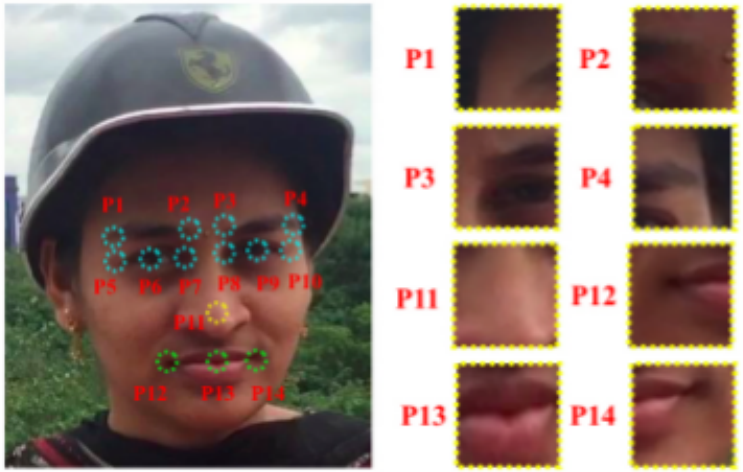


Figure 1: The figure (left) illustrates the 14 facial key-points annotated for both the introduced datasets. The description of the facial points is as: Eyes region (cyan): P1- left eyebrow outer corner, P2- left eyebrow inner corner, P3- right eyebrow inner corner, P4- right eyebrow outer corner, P5- left eye outer corner, P6- left eye center, P7- left eye inner corner, P8- right eye inner corner, P9- right eye center, P10- right eye outer corner; Nose region (yellow): P11- nose; Lip region (green) P12- lip left corner, P13- lip centre, P14- lip right corner. Few key points have been shown on the right.

Figure 2: The illustration shows samples images with different disguises from both the Simple and Complex face disguise (FG) datasets. As seen from the image, the samples from the complex background dataset have a relatively complicated background as opposed to the simple dataset.

data. However, such datasets are not available (small:

On arxiv since late August, to be published at ICCVW 2017. It is unclear how much performance can be improved.

Disguised Face Identification

55% accuracy on easier dataset

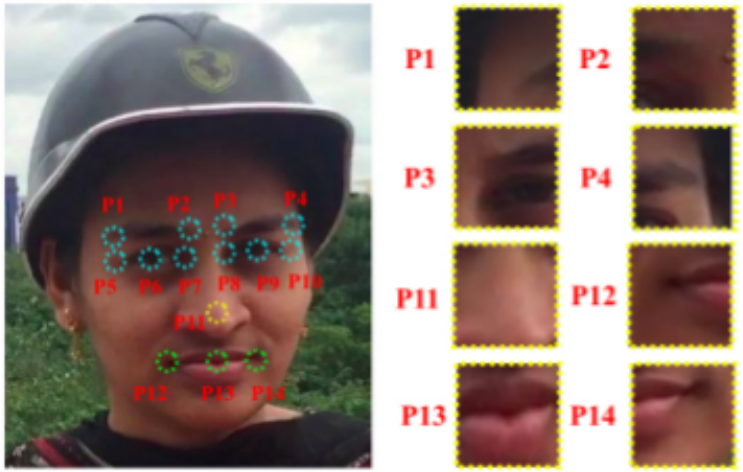


Figure 1: The figure (left) illustrates the 14 facial key-points annotated for both the introduced datasets. The description of the facial points is as: Eyes region (cyan): P1- left eyebrow outer corner, P2- left eyebrow inner corner, P3- right eyebrow inner corner, P4- right eyebrow outer corner, P5- left eye outer corner, P6- left eye center, P7- left eye inner corner, P8- right eye inner corner, P9- right eye center, P10- right eye outer corner; Nose region (yellow): P11- nose; Lip region (green) P12- lip left corner, P13- lip centre, P14- lip right corner. Few key points have been shown on the right.

Figure 2: The illustration shows samples images with different disguises from both the Simple and Complex face disguise (FG) datasets. As seen from the image, the samples from the complex background dataset have a relatively complicated background as opposed to the simple dataset.

data. However, such datasets are not available (small:

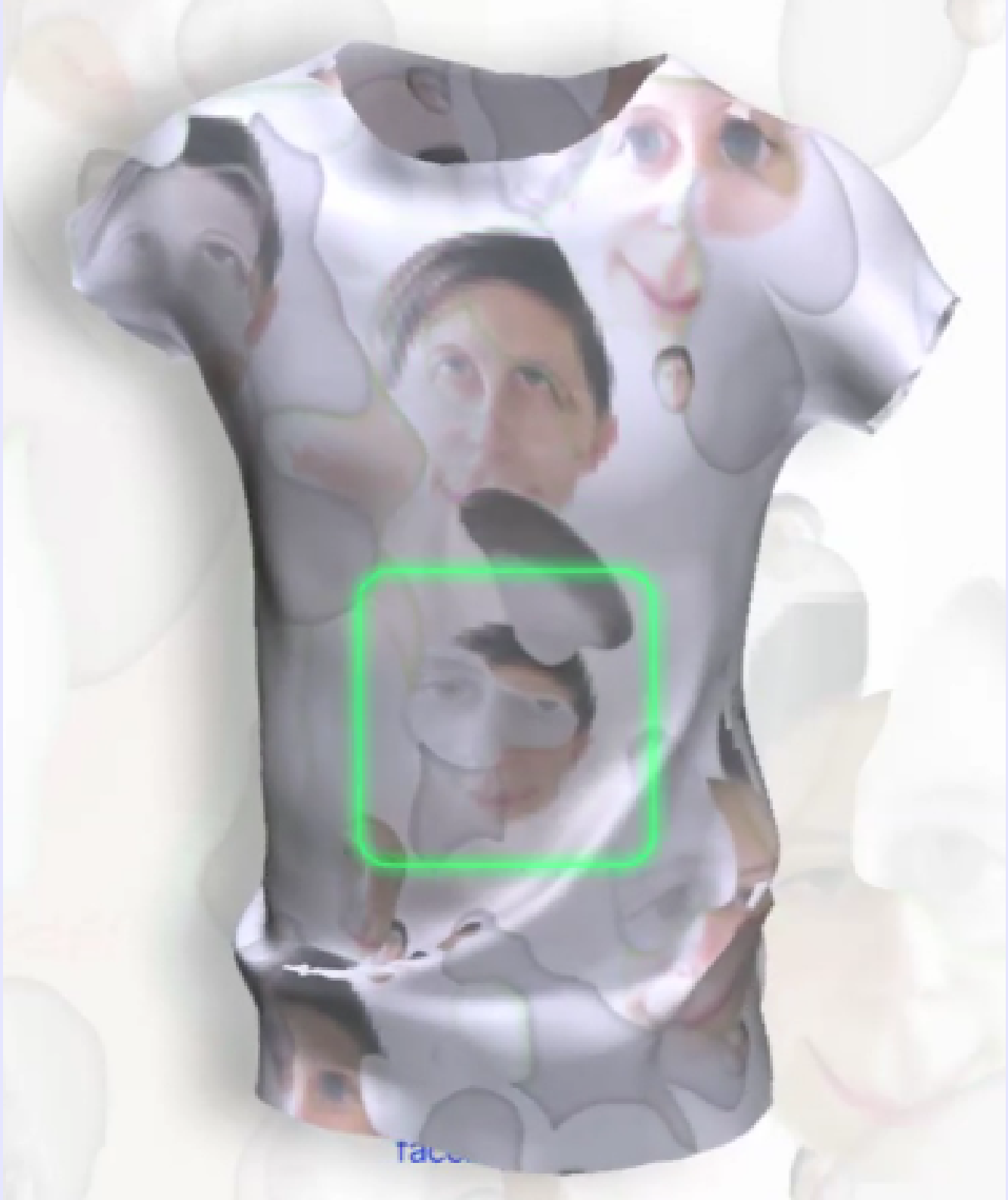
On arxiv since late August, to be published at ICCVW 2017. It is unclear how much performance can be improved.

Obfuscation

- “If privacy both as impossible-to-observe and impossible-to-identify is dead, then what might be an alternative? If you’re an optimist or an apparatchik, your answer will tend toward rules of procedure administered by a government you trust or control. If you’re a pessimist or a hacker/maker, your answer will tend toward the operational, and your definition of a state of privacy will be mine: the effective capacity to misrepresent yourself.”

  Geer, ‘Identity as Privacy’


Michael Jackson
Michael Jackson
Michael Jackson
Michael Jackson
Barack Obama
Barack Obama
Barack Obama



Another approach: make more video collectors trustworthy

- Personalized Privacy Assistants
  - A registry of IoT devices that respects users requests about how their data will be handled
  - https://youtu.be/j5btHZKgwaI
- A good idea for those willing to participate in the registry (which is probably a lot of device owners, and participation could be made compulsory in some contexts)
- No good if the device owner spurns the registry, or if they try to use it to mislead